UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Temporización mediante el temporizador del sistema SysTick en microcontroladores ARM Cortex-M

Apellidos, nombre	Capella Hernández, Juan Vicente (jcapella@disca.upv.es)
Departamento	Departamento de Informática de Sistemas y Computadores
Centro	Universidad Politécnica de Valencia

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

1 Resumen de las ideas clave

En este artículo se introduce la temporización mediante el temporizador del sistema *SysTick* en microcontroladores ARM Cortex-M. Se realiza una explicación de sus fundamentos, así como de su implementación paso a paso, de forma que se pueda comprender su funcionamiento, y aprender su configuración básica, sentando las bases para poder avanzar en la implantación de aplicaciones para esta familia de microcontroladores.

2 Introducción

Muchas aplicaciones de sistemas empotrados requieren realizar retardos de precisión o tareas periódicas (Semáforos, pasos a nivel,..) o por ejemplo generar señales como ondas cuadradas, siendo para todas estas tareas necesaria llevar a cabo una temporización exacta.

Esta temporización puede ser llevada a cabo por software, para lo cual es necesario conocer cuánto tardan en ejecutarse las instrucciones. Ahora bien, esta aproximación tiene diversos inconvenientes, como no poder trabajar con frecuencias altas, además si hay interrupciones es imposible lograr precisión. Por otro lado con esta aproximación la CPU desperdicia su tiempo y no puede hacer otras cosas. Por todo ello, la solución pasa por emplear los temporizadores que integran los microcontroladores y permiten realizar temporizaciones exactas y liberar de estos menesteres a la CPU (funcionan en paralelo con la CPU).

En este artículo se pretende introducir la temporización mediante el uso del temporizador de sistema SysTick en los microcontroladores ARM Cortex-M4 STM32 [1], ofreciendo algunos ejemplos de código, utilizándose el entorno de desarrollo Keil.

3 Objetivos

Una vez que el alumno se lea con detenimiento este documento, será capaz de:

- Conocer y comprender la utilidad de los temporizadores.
- Sintetizar programas que hagan uso del sistema de temporizadores.
- Sintetizar manejadores de interrupción para los temporizadores

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

4 Desarrollo

A continuación se desarrollarán cada uno de los aspectos indicados en la introducción y objetivos, realizando las explicaciones de la forma más práctica y guiada posible.

4.1 SysTick

Se trata de un temporizador de 24 bits de cuenta descendente, que produce una interrupción cuando el registro interno llega a cero desde el valor de recarga inicial [2]. En la figura 1 puede observarse el mapa de registros de este temporizador.

El flag COUNTFLAG se pone a 1 cuando el contador pasa del valor de cuenta 1a 0. Para que vuelva a 0 se puede leer el registro de control de SysTick, o poner a 0 la cuenta escribiendo cualquier valor en el registro de cuenta.

Offset	Registro	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	**STK_CTRL** *Valor de reset*	Reservado															COUNTFLAG 0	Reservado													CLKSOURCE 1	TICKINT 0	ENABLE 0
0x04	**STK_LOAD** *Valor de reset*	Reservado								RELOAD [23:0]																							
										0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	**STK_VAL** *Valor de reset*	Reservado								CURRENT [23:0]																							
										0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	**STK_CALIB** *Valor de reset*	Reservado								TENMS [23:0]																							
										0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figura 1. Mapa de registros SysTick

Su mayor ventaja es su sencillez, dado que no hay que configurar nada excepto el valor de recarga citado anteriormente. El inconveniente es que al no poder configurar preescalas ni otros parámetros las temporizaciones que se pueden realizar son bastante básicas.

4.2 Configuración del SysTick

Para configurar el SysTick deben seguirse los siguientes pasos:

- Desactivar el contador. ENABLE=0.
- Cargar el valor de RELOAD.
- Escribir cualquier valor en la cuenta para que se ponga a 0.
- Configurar los registros de control y estado, incluyendo la activación.

Esta funcionalidad es proporcionada por la función `SysTick_Config()`, que configura y pone en marcha el temporizador SysTick.

Concretamente, esta función CMSIS (Cortex Microcontroller Software Interface Standard) configura y realiza las siguientes funciones:

- Configura el registro de recarga del SysTick con el valor pasado como parámetro
- Configura la prioridad de la interrupción del SysTick al valor más bajo (IRQ priority).
- Configura la fuente de reloj para el contador del SysTick a HCLK - Core Clock Source.
- Habilita la interrupción del timer.
- Inicia el contador.

Para ajustar el tiempo base del SysTick se utiliza la siguiente fórmula:

Valor de recarga = SysTick Counter Clock (Hz) x Temporización deseada (s)

El SysTick Counter Clock es el reloj que le llega al temporizador SysTick, que en el STM32F4 es 168 Mhz.

El valor de recarga es el parámetro que le pasamos a la función SysTick_Config(), que no debe exceder 0xFFFFFF.

Si fuera necesario, se puede cambiar la prioridad del temporizador usando la función NVIC_SetPriority(SysTick_IRQn,...) después de invocar la función SysTick_Config(). La función NVIC_SetPriority() está definida en el fichero core_cm4.h

4.2.1 Ejemplo

En este ejemplo veremos cómo medir el tiempo, sabiendo la velocidad del reloj y realizando los cálculos indicados.

Ejemplo:

Reloj de 168 MHz y queremos medir 1 ms.

RELOAD = 168000000×10^{-3}

4.3 Implementación del manejador de interrupción asociado

Una vez configurado el temporizador, solo nos queda implementar el manejador de interrupción asociado. Dicho manejador es el `void SysTick_Handler(void)`:

```
void SysTick_Handler(void)
{

    // aquí el código de la rutina de servicio del timer

}
```

En dicha rutina programaremos aquello que deseemos ejecutar cuando termine la temporización configurada del SysTick.

4.4 Ejemplo completo

Para ejemplificar el uso del temporizador SysTick se proporciona a continuación un ejemplo que pretende generar una onda cuadrada de frecuencia 500 Hz por el pin PD12:

```
#include "main.h"

GPIO_InitTypeDef GPIO_InitStructure;


int main(void)
{

  /* Inic. el Led4 montado en la STM32F4-Discovery board – conectado al PIN PD12*/
  STM_EVAL_LEDInit(LED4);


  // La señal a generar es de 500Hz: el periodo de la misma será 2 ms
  // Inic. SysTick Timer para que salte la interrupción cada 1 ms (semiperiodo)

 SysTick_Config(SystemCoreClock / 1000);
```

```
  while (1)
  {

    // Bucle vacío ya que la funcionalidad está en el manejador de la interrupción
    // asociada al temporizador SysTick.

  }
}
```

En el fichero stm32f4xx_it.c programaremos el manejador de la interrupción asociada al SysTick:

```
/**
  * @brief  This function handles SysTick Handler.
  * @param  None
  * @retval None
  */

void SysTick_Handler(void)
{

  // Cada semiperiodo cambiamos el nivel de la salida:
  // si estaba a "1" pasa a "0" y viceversa, generando la señal cuadrada deseada
  STM_EVAL_LEDToggle(LED4);

}
```

5 Cierre

A lo largo de este objeto de aprendizaje hemos tratado los fundamentos de la temporización mediante el temporizador del sistema SysTick de los micrcontroladores ARM Cortex-M. Además se han proporcionado un ejemplo que sirva de base para aprender a configurarlo y usarlo en otras aplicaciones.

Para comprobar qué realmente has aprendido las bases de la temporización, es el momento de que te pongas manos a la obra e intentes crear un proyecto que configure y utilice el SysTick, y observar su funcionamiento. Es como mejor se aprende. Por ejemplo, calcula el RELOAD para medir 10 ms. También puedes calcular el tiempo máximo que podría medir el timer.

¡¡ÁNIMO!!

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

6 Bibliografía

[1] ARM Limited. Cortex-M4 technical reference manual, 2010. URL: http://www.arm.com/

[2] J Yiu: The Definitive Guide to ARM® Cortex®-M3 and Cortex-M4 Processors, 3rd Edition, 2013.

[3] Microelectronics, St. STM32F3xxx and STM32F4xxx Cortex-M4 programming Manual, 2013. URL: http: //www. st.com/web/en/resource/technical/document /programming_manual/DM00046982.pdf